Neuro-Symbolic Robotics

Emre Ugur[1], Alper Ahmetoglu[2], Yukie Nagai[3], Tadahiro Taniguchi[4], Matteo Saveriano[5] Erhan Oztop[6]

***Abstract*—This paper presents and gives an overview of the recently emerging field of Neuro-Symbolic Robotics. With the advances in computational resources, robust neural architectures, and big data, Neural Networks have become the natural solution to all robotic problems that require emergence, learning, adaptation, and, recently, reasoning and communication. However, to ensure the safe deployment of robots in the real world, they lack crucial properties like verifiability, explainability, and interpretability. Additionally, neural network-based systems suffer from generalization and extrapolation problems, which limit their scalability. Symbolic systems have been used since the birth of intelligent robotics, offering verifiability, explainability, and scalability; however, their manually coded implementations could not cope with the richness and wide varieties of the continuous and high-dimensional world of the robots. In this paper, we review the robotic architectures that integrate neural networks and symbolic systems in different ways, benefiting from their advantages. We categorize the robotic systems into four broad categories, namely, intertwined, coupled, non-uniform neuro-robotic systems, and neuro-symbolic translation, discussed in detail the capabilities and limitations of these systems and discussed the future challenges in this field.**

***Index Terms*—Neuro-Symbolic AI, Neuro-symbolic Artificial Intelligence**

I. Introduction

Intelligent Robotics can be defined as the intersection of artificial intelligence and robotics, aiming to build machines capable of learning, reasoning and performing complex tasks autonomously in dynamic environments. The key challenges in intelligent robotics include learning robust yet flexible representations that facilitate not only complex task execution but also reasoning for safety and explainability.

In contrast to domain-specific machine learning problems such as classification and regression, a robot needs to process a continuous stream of sensorimotor data but yet act on a world that is structured as a network of discrete entities with a range of relations among them. Thus, intelligent robots need cognitive mechanisms to work with continuous sensory input and abstract symbolic structures [1]. In the following, we first address them separately under *Learning Systems* and *Symbolic Systems* and later discuss how the two worlds can be intertwined together.

Learning Systems. The groundbreaking advances in deep neural networks and their effective application in artificial intelligence raise the question of whether deep learning with big data is the solution that Intelligent Robotics is seeking. The recent state-of-the-art robotic learning studies employ some form of (deep) neural network architecture, delivering superior performance compared to earlier traditional machine learning methods. Yet, the lack of transparency in neural networks poses serious concerns about their reliability, robustness, and safety [1].

Another significant criticism pertains to the brittleness (being susceptible to adversarial attacks) of the models [2], as well as the data-efficiency and computational expense associated with robotic implementations [3]. While the computing cost can potentially be mitigated by suggesting a central pre-training that is performed once and subsequently deployed across various locations with minimal or no further training, addressing the black-box issues remains challenging, because post-hoc explanations of neural network outputs may lack the reliability needed to persuade end-users to integrate these technologies into actual robots. For example, although recent large language model (LLM) based systems allow pseudo-reasoning, their reasoning capabilities are not verifiable or reliable [4], even though they may be optimized through data fine-tuning and/or reinforcement learning with human reward labeling for valid chain-of-thought generation. Thus, in intelligent robotics, the reasoning capabilities of these systems are limited to constrained laboratory settings.

Computationally, neural networks can represent propositional logic and a restricted subset of first-order logic, but cannot represent full functionality and representational power of first-order logic, according to Marcus 2020 [3]. Thus, albeit the impressive reasoning-like abilities of LLMs it is not clear how formally defined computational semantics can be neurally embedded in the operation of LLMs to address the questions of reliability, trustworthiness and safety in robotics.

Symbolic Systems. Symbolic systems are reliable in terms of planning and reasoning abilities, as computation steps can be explained and proved for correctness [5], [6]. However they lack flexibility [7], for example an execution plan can be correct but still may fail in a given environment setting if the symbols used to capture the current setting lack the required resolution or sensitivity. This is a classical example of the problem of pre-defining a set of symbols and rules to represent the sensorimotor experience of a robot, which is often called the *symbol grounding problem* [8]. Regardless of how well a symbolic system may be designed, it inevitably becomes fragile when confronted with minor alterations in the embodiment, environment, or task that were not anticipated during the design phase. Another issue with symbolic systems is that they allow reasoning only in symbolic space. However, the robotics tasks of planning, monitoring, and validation may require representations at multiple levels of abstraction beyond a single symbolic level. Although the levels of abstraction that are used in robotics literature are delineated in [7] within the context of natural language representation, a theory of multi-

This work was supported by the INVERSE project (101136067) funded by the European Union.

[1]Bogazici University [2]Brown University 32The University of Tokyo [4]Kyoto University [5]Trento University [6]Osaka/Ozyegin University

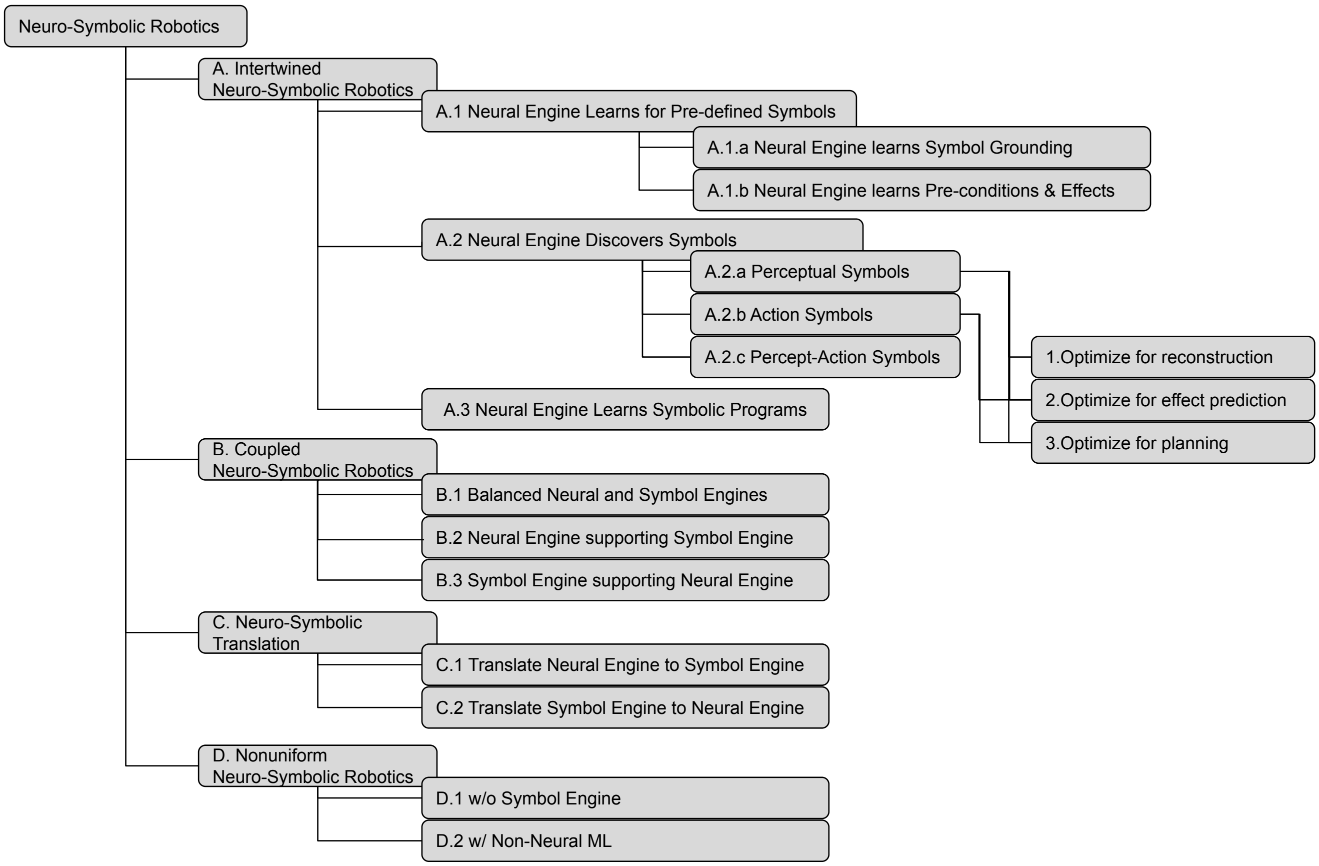

Fig. 1. A taxonomy of Neuro-Symbolic Robotics

level symbolic manipulation bridging the low-level sensory input with higher-level representations is lacking [2].

Neuro-Symbolic Systems. With the recent advances in deep neural networks, now it has become more possible to address the symbol grounding problem by letting advanced neural network architectures learn symbolic representations. These representations can be not only used in symbolic manipulation systems for reasoning and planning, but also equip the robot with different capabilities, or improve the existing ones.

Although there are very preliminary efforts for proposing general paths for adapting neuro-symbolic approaches into, for example, industrial [9], surgical [10] or assistive [11] robotics, a general overview of the current studies and a possible generic architecture that utilizes full-power of both neural and symbolic systems are still missing. In the remaining of the paper, we systematically analyze key robotic works from the literature that have used symbolic and learning systems together with a varying extent. The analysis is guided by our proposed taxonomy of robotic studies that have exploited both symbolic and learning systems ranging from loosely coupled ones to tightly coupled neuro-symbolic robotic systems.

II. Definitions

Symbol Representation corresponds to encoding of robot perception, action or state in discrete space. **Continuous Representation** corresponds to continuous encoding of robot perception, action or state, that might be used as input and/or output of a Neural Network system.

Symbol Engine corresponds to the methods and algorithms used for manipulating symbols. It might corresponds to classifiers such as decision trees, Monte-Carlo search trees used for multi-step prediction, operations over Domain Specific Language (DSL) or full-fledged off-the-shelf AI planning in standard symbolic languages such as Planning Domain Description Language (PDDL) [12].

Neural Engine corresponds to Neural Network used in discriminatory or generation tasks. The inputs and outputs of the Neural Engine, as well as the intermediate representations might be symbolic or continuous, depending on the task.

III. A. Intertwined Neuro-Symbolic Robotics

In this category, the representations, rules, or programs used by the Symbolic Engine are generated by the Neural Engine. A key distinction among these types of approaches is whether program generation is at the core, or the symbol discovery is undertaken by the neural system or not. Accordingly, we have three main subcategories which are detailed next.

A.1 Neural Engine Learns for Pre-defined Symbols

In this category, the symbols and the operators used by the Symbol Engine are pre-defined based on the task and domain requirements. The Neural Engine either learns the mapping between these discrete symbols and the continuous sensorimotor experience of the robot or the set of symbolic pre-conditions and effects of the operators from the robot's experience.

A.1.a Symbol grounding: For constrained domains and tasks, planning is possible with pre-defined sets of predicates, operators, and pre-conditions and effects of these operators. In these situations, the representational gap between the symbols and the continuous representation the robot faces should be addressed. For this, given pre-defined predicates in pre- and post-conditions of manually designed transition rules, the robot learned the mapping from its own percepts to the corresponding predicates post-conditions using kernel perceptrons [13]. More recently, [14] learned to process RGB image patches conditioned on canonical object views into embeddings that can be classified to single and relational object-object logical predicates encoded in action preconditions from demonstrations to be used in planning. Given a robot interaction video dataset with annotated actions and manually implemented pre-conditions and effects, [15] trained classifiers that map the bounding boxes of objects to the corresponding symbols. However, the long-horizon planning was left as future work. When applicable, the approach taken in this category is effective, but the fixed set of predicates and pre-defined rigid rules and operators make the practical deployment of the method limited to well structured environments and tasks.

A.1.b Pre-condition and effect learning: The rigidity in the pre-defined transition rules used can be relaxed by learning the set of predicates for the pre-conditions and post-conditions of the operators used by the Symbol Engine. In this category, the mapping between the sensorimotor space of the robot and the symbols has been established before, and the robot basically learns the set of predicates for pre-conditions and effects of actions. [16] learned a list of pre-condition and effect predicates, first segmenting the human demonstrations into actions, then extracting the relevant pre-conditions and post-conditions based on counting heuristic, and finally generating the related planning operators. In the end, an externally given goal can be satisfied by a sequence of actions using Fast Downward PDDL planner [17]. [18] learned action pre-conditions and effects in the form of lists of symbolic predicates from provided human demonstrations in manipulation domain and verified through PDDL planner. Given pre-defined symbolic predicates, [19] learns a set of parameterized actions, with their corresponding pre-condition and effect predicates in a manipulation domain. [20], on the other hand, proposed to learn the planning domains from the observed traces using Behavior Trees as intermediate human-readable structures. Given a symbolic goal, [21] learned the necessary symbolic operators to be able to synthesize a plan and its low-level controller implementation in an RL framework. Following a different approach, [22] used preconditions and effect symbols to detect task-specific deficiencies and support humans in action feasibility, rather than plan generation. These studies are constrained with pre-defined set of symbols, which will be addressed in the next section.

A.2. Neural Engine Discovers Symbols

In this section, we cover the studies where Neural Engine is used to discover perceptual (e.g. [23]), action (e.g. [24], [25]) or sensorimotor (e.g. [26]) symbols that are directly used by the Symbol Engines for planning and for other purposes.

A.2.a Emergence of perceptual symbols: Neural Engines, in this category, find/form/discover symbols [27] in the continuous sensory/perceptual space of the robot. These symbols are typically employed as predicates in the preconditions and post-conditions of action operators in planning. These studies optimize the process of the organization of the continuous perceptual space to find discrete symbolic categories using different approaches and metrics as follows.

a) A.2.a.1 Optimize for reconstruction: The most straightforward way to find discrete categories from continuous representation is to apply unsupervised clustering algorithms. In this way, each cluster would correspond to a symbol. Early studies such as [28] used probabilistic generative models, such as Latent Dirichlet Allocation (LDA), to learn multi-modal symbols from sound and observation of objects during physical grasp actions of the robot and from provided words. In a follow-up study [29], the robot acquired object concepts utilizing the word sequences, which are segmented in an unsupervised way, in addition to the multimodal information obtained from object observations. In [30], the online version of this algorithm is provided. In [31], a similar unsupervised multimodal categorization method (probabilistic Latent Semantic Analysis) that used haptic, audio, and visual information was proposed. In these experiments, a robot interacted with objects through various actions, listening to the sound and also observing the object from different viewpoints. [32] discovered object categories and multi-modal symbols using modal latent Dirichlet allocation (MLDA) and variational auto-encoders. [33] extended this approach to symbol formation through interpersonal interactions. These methods were used for probabilistic inference rather than long-horizon planning. For example, [34] studied whether a deep reinforcement learning system could entail the development of high-level neural encodings that might be viewed as antecedents of symbolic representations. They showed that even without explicit design or engineering, neural responses that resemble abstract symbol-like representations might emerge in their system. Recently [35] proposed first discovering skill segments from demonstration trajectories, then applying unsupervised clustering and SVM classification to identify and learn the mapping of the potential termination states of each learned skill, and finally learning the relation between natural language sentences and sequences of the learned abstract symbols using Seq2Seq recurrent neural network.

A.2.a.2 Optimize for action effect prediction: The unsupervised clustering approach finds discrete symbols, without any guarantee to be useful for the Symbol Engine. To address this problem, a number of research groups investigated how to discover symbols that are guaranteed to be useful for the most basic step of symbolic planning, for one-step action effect prediction.

[23] proposed and realized a general neural framework that translates the robot's raw sensorimotor experience into

the symbolic domain. With this architecture, given continuous interaction experience, the robot can discover object and effect symbols that can be automatically translated to Probabilistic Problem Domain Definition Language for making symbolic high-level plans. A predictive deep encoded-decoder network with a binary bottleneck layer was trained with initial and outcome scene images to extract action and effect-grounded object and outcome categories, which in turn were used to make single-step action predictions. As an intermediate step, a decision tree was trained from robot's interactions, now represented with the corresponding symbolic object and effect categories. Each path in the decision tree was converted to an operator in PPDDL format, allowing the use of the off-the-shelf planners as Symbol Engines. The system was verified in a table-top environment, where symbols such as pushable, rollable, or insertable were discovered and used to make effective plans, for example, to build towers from objects at different heights. [23] could learn symbols only for single or paired objects, and the preconditions and effects in the planning operators include single or paired object predicates. On the other hand, many complex actions involve interactions with varying numbers of objects, or the effects of actions influence multiple objects in complex environments such as cluttered or articulated settings. [36], [37] extended DeepSym by adding a transformer structure to learn the attention weights of object features. These weights again go through a discrete activation layer, which generates the relational symbols. The relational symbols between objects are combined with the discovered single-object symbols to predict the outcomes of the actions of the robot for each object. The continuous interactions are converted to symbolic interactions using this attention mechanism and later transformed to PDDL operators in [38]. Their system was again verified in a table-top setting, showing that the system can use off-the-shelf AI planners to generate plans that require the use of symbolic operators with multiple objects. However, the objects did not have complex shapes in this study, loosely addressing object affordances.

A.2.a.3 Optimize for planning: The symbols that are effective in single step action prediction do not necessarily guarantee to be optimal in their planning performance. Therefore, the most recent studies in the Symbol Discovery category aimed to discover symbols to maximize their planning performance. [39] learned abstract relational symbolic object representations from raw visual observations in an unsupervised way and used them to make multi-step plans. The groundings were evaluated and refined in the rollouts of a planning loop. Neural Engine output was used as the parameters of the probabilistic models, which in turn can be used as the Symbol Engine. The model was used for model-based RL in simulated tower-building tasks for simple blocks, given images of goal and initial blocks.

A.2.b Emergence of action symbols: The previous studies where perceptual symbols were discovered assumed the existence of discrete actions. However, a truly lifelong cognitive robotic system should have the capability to learn discrete actions as well [40]. In this section, we review the studies where action symbols were discovered and used in Symbol Engines.

A.2.b.1 Optimize for action effect prediction: [25] proposed formalizing operator learning problem in task and motion planning (TAMP) framework, where their system learns operators on previously acquired symbols which can be defined as lossy abstractions of the transition model of a domain. Followinge [25], [41] proposed neuro-symbolic relational transition models (NSRTs) in which a task plan skeleton is generated using a symbolic engine that describes the high-level transitions, and then the neural engine searches for low-level operator parameters. If the plan skeleton is not *downward refinable*, i.e., if there is no parameterization of the lower-level skill that makes the plan successful, a new plan skeleton is generated, providing *bilevel* planning in both discrete and continuous levels, which allows the robot to make detailed plans considering the geometric information. NSRTs are learned from a given set of parameterized skills. [42], on the other hand, learns these skills as well from low-level demonstrations, providing a complete bilevel operator learning stack. [43], in a hierarchical RL framework, discovered a diverse set of actions and simultaneously learned symbolic forward models through intrinsic motivation signals given predefined state abstractions. As Symbol Engine, the system used the breadth-first search method where each expansion corresponded to the learned symbolic forward model and executed one-by-one in order to reach the goal. [44], [45] proposed to learn constraints that address the effectiveness of actions using Gaussian Processes. They proposed a sampling method for creating a rich set of potential action parameters along with the skills. Given a goal and learned parametric motion primitives, the planning system receives perceptual state estimates from the Neural Engine to generate a plan using their so-called PDDLStream framework [46], [47]. Last but not least, [48] discovers action symbols from human demonstrations and exploits VLMs not only to label those actions but also to generate plans through their scene interpretation and reasoning capabilities.

A.2.b.2 Optimize for planning: While many previous approaches either used unsupervised clustering or self-supervision based on single-step effect prediction to learn predicates, [49] learned symbolic predicates with a surrogate objective for multi-step planning. They used interactions obtained from demonstrations rather than the robot's own self-exploration experience of the world. In follow-up work, [50] reduced the complexity of the learned operators by focusing on a subset of abstract effects. These studies not only learn action symbols but also find the motion parameters that would allow task and motion planning (TAMP). Although symbolic predicates are learned as well, these are defined over already available high-level predicates, which might not be realistic to assume in life-long scenarios. Following [49], [51] proposed to learn predicates by actively collecting information by querying an expert. [52] jointly learned a set of symbolic action abstractions and their low-level controllers utilizing LLMs in an interactive planning loop.

Studies mentioned in this section either assume an existence of high-level predicates or a set of demonstrations from which a good set of state symbols can be learned. As such, the low-level policies of operators are learned with supervision, either in the form of state symbols or demonstrations that solve the task. It is also worth mentioning option discovery methods [53]–[57] that focus on learning these low-level policies directly from raw sensory space by exploration. While these methods do not directly use a symbol engine, they provide a finite number of low-level policies with their initiation and termination conditions overlapped in the state-space, which, in essence, bootstraps most of the state-symbol learning methods.

A.2.c Emergence of Percept-Action Symbols: While the previous studies focused on discovering either perceptual or action symbols, some recent studies addressed the challenge of discovering both perceptual and action symbols from the sensorimotor experience of the robot. [26], [58], [59] used critical regions [60], [61], which are high density parts in the state-space, as state abstraction targets, and learned action abstractions on top of them. The provided low-level demonstrations, which are generated by motion planners, defines a density in the state-space. More recently, [62], [63] extended [49] by first learning high-level predicates directly from raw state representations using visual language models (VLMs) and then learning operators defined over these predicates.

A.3 Neural Engine learns Symbolic Programs

In this category, the complete program, processed by the Symbol Engine, is generated by the Neural Engine, mostly by the Large Language Models (LLMs). In [64], a Neural Engine (LLM) generated parallel plans through a sequence of transformations, which were translated into the behavior trees and executed by the robot. Here, the Neural Engine was used to transform symbols (in natural language) into Prolog programs first and behavior trees later. [65] trained LLMs to produce neuro-symbolic task planners, which are consistent with PDDL. With this, they obtained better scalability when the domain complexity was increased. This approach also enabled producing actions without waiting for the generation of the whole plan. In [66], a pre-trained Neural Engine (LLM) was used to learn symbolic predicates, in the form of Python program segments, from human language feedback during robot interactions. Next, symbolic operators were learned through a clustering algorithm, enabling plan generation. Last but not least, [67] provided interpretability in symbolic decision-making in autonomous driving by combining neural and symbolic approaches, also achieving safe and stable behavior. A Neural Engine was trained to select operations from a set of symbolic pre-defined operations. This allowed generating a sequence of operations given goals in Domain Specific Language (DSL), which was consumed by the Symbol Engine for planning and control. With the recent advanced in LLMs, we expect to see more studies in this category.

IV. B. Coupled Neuro-Symbolic Robotics

In this section, we overview the robotic systems where Neural Engine and Symbol Engine modules interact with each other, by combining their outputs or by supporting the each other.

B.1 Balanced Neural and Symbol Engines

Say-Can [68] has been one of the first studies that benefited from the reasoning capabilities of LLMs for robot control. For this, given a goal, they used a language model (PALM [69]), which provides high-level semantic knowledge about the task and provides a list of actions to achieve the corresponding task. In order to ground the corresponding actions in the actual world of the robot, an affordance-based value function module was implemented, which was used to weigh and filter the actions produced by the language model. This method can be considered to be the first LLM based model capable of completing long-horizon natural language instructions on a manipulator with mobility. Neural and Symbol Engines were used as building blocks in [70], which presents a modular approach where action primitives are defined to handle independent subtasks. The input query was processed by a language parser, transforming it into an executable program composed of such primitives. Note that some primitives were symbolic (e.g. counting), and others were implemented with neural networks (e.g. visual grounding). [71] used a Symbol Engine to select a list of safe actions in each RL exploration step, and a Neural Engine, which approximates the Q value function, to select the action to execute following the learned policy. With this, they ensured safety and also enabled control in the continuous state and action space. [72] integrated logical rules, ontologies, and LLM-based planners, and exploited symbolic information to improve the ability of LLMs to generate recovery plans. Given an instruction, their robot started executing the actions for the plan generated by LLM. The effect of each action was observed, and provided as input to the sub-goal verifier, which used an ontology and decided whether the action was successful or not. In case of failure, the ontology was again used to decide the recovery strategy which was provided to the LLM-planner for replanning. In [73], human instructions were translated into executable robot plans by using LLMs to decompose the tasks into sub-goal descriptions that were executed by the planner sequentially. They use scene graphs as the intermediate representations. [74] combined symbolic and geometric scene graphs for vision-based long-horizon hierarchical planning. A symbolic scene graph is used to find the next sub-goal from the goal description, and the geometric scene graph is used to predict the motion parameters.

B.2 Neural Engines supporting Symbol Engines

In this category, the main control is on the Symbol Engine, and the Neural Engine is used to support the Symbol Engine in executing the tasks. [75] proposed to train neural network classifiers to forecast the viable motions and employ the classifier as a learned heuristic, steering the TAMP search toward possible motions and decreasing the total amount of motion planning trials. [76] proposed a Neural Engine that uses an initial image of the environment, predicting the promising sequence of discrete actions providing runtime improvements of several magnitudes. Given expert demonstrations, [77] applied learning techniques to efficiently search in the high-level task planning space, taking into account the possible infeasibilities and, as a result, significantly increasing the planning speed of the Symbol Engine. [78] proposed a learning system that improves symbol grounding functions and a high-level planning method to optimize the total performance of the existing hierarchical planner in generating suitable plans. [79] learned relational state representations, transition function, grounding function, and action-value functions to support the planning experience of a robot. PDDL planning was used in the exploration of the agent. Learning allowed the agent to scale up larger environments. [80] used the predicted confidence values from a Neural Engine to infer probabilistic belief states that were used by the Symbol Engine.

B.3 Symbol Engines supporting Neural Engines

In this category, the Symbol Engine is used to support the Neural Engine, which acts as the main controller. [81] used the principles of maximum information compression and slowly varying signals to extract symbol-like representations that enable fast skill transfer. The activations on the the last hidden layers of the Neural Engines were used for this purpose. While the symbols enabled fast transfer, an explicit Symbol Engine was not fully utilized in this work. [82] leveraged the symbolic representation from the high-level planner to direct trial-and-error-based skill learning. Their system learns temporally-extended actions to achieve the desired outcomes of the symbolic operators by using a reward taking into account the post-condition of the operator within the Reinforcement Learning loop. [83] also proposed a method that used Symbol Engine to decide exploratory actions for training the Neural Engine in a simulated mobile robot. [84] proposed to use PDDL Symbol Engine to improve the neural perceptual capabilities of the agent.

V. C. Neuro-Symbolic Transformation

C.1 Transform Neural Engine to Symbol Engine

The robots controlled by Neural Engines generally lack explainability, interpretability, and verifiability, as we discussed in the Introduction section. In order to address this problem, some studies transformed the policies encoded by Neural Engines into symbolic representations. For example, [85] proposed an algorithm for learning a range of comprehensible skills with their parametric representations derived from the planning strategies of an agent. For explainability and verifiability, [86] trained verifiable policies encoded with decision trees and realized their framework in the cart-pole task in an RL setting. Given a trained RNN, [87] learned the abstraction and extracted a deterministic finite automaton that encodes the state dynamics of the task For transparency, trust, explainability, and interoperability, [88] applied clustering in the latent space of the internal states. Using the hidden states, they generated a finite-state automaton that captured the underlying grammar, enabling the prediction of whether a given pattern is valid or not. In order to ensure compliance with behavioral specifications through formal guarantees, e.g. safety and/or reachability, [89] proposed a method to autonomously build a finite-state machine from a recurrent neural network, accommodating existing formal verification tools in agent benchmarks.

C.2 Transform Symbol Engine to Neural Engine

In this category, the manually encoded symbolic policies are transformed into continuous neural policies and refined through the robot's experience. [90] incorporated a Symbol Engine, which used linear temporal logic (LTL), into the training of a Neural Engine such that each neural network in the system corresponded to a particular symbolic representation. The resulting Neural Network-based planner in this model inherited the symbolic model's interpretability and correctness assurances, with the aim of generalization to unseen tasks, including new workspaces, novel temporal logic formulas, and errors in the robot's dynamical model. [91] used a Symbol Engine to realize symbolic policies and another Symbol Engine for formal verification for safety in every exploration loop of an RL-based robot learner. In their work, verifiable safe symbolic policies were transformed into continuous policies realized by the Neural Engine, updated using reward-based gradient updates, and transformed back to the original symbol space.

We expect an increase in the adoption of the methods that transform one engine to the other one in more robotic tasks to ensure verifiability and interoperability.

VI. Nonuniform Neuro-Symbolic Robotics

The studies use both continuous and symbolic representations. However, the full-fledged power of one of the Symbol or Neural Engines is missing.

w/o Symbol Engine

In this category, Neural Engines are used to process (as input and output) symbolic and continuous representations for task understanding and robot control. However, these symbolic representations are not exploited by the Symbol Engines.

[92] extended [93] to robotics domain and proposed a "Neuro-Symbolic program" which process both continuous and symbol representations using trained Neural Engines. It trains a Neural Engine to parse natural language instructions, transforming it into a program in Domain Specific Language (DSL), and another Neural Engines such as a Visual Extractor receives the visual scene and produces visual features. The DSL description and visual features are combined as inputs in the Neural Engine Visual Reasoner that outputs an action. The framework is trained end-to-end. No Symbol Engine is used in this approach.

LLM based robot control frameworks such as RT-1 [94], RT-2 [95], and RT-X [96] bootstrap their model with pre-trained foundational models, and use the multi-modal experience of multiple robots in different environments in order to learn robot controllers that can execute plans generated by the LLMs. Given different goals and visual scene descriptions, these LLM-based systems can both generate a chain of actions and robot control commands, such as the target displacement of the robot's gripper at each step. [97] used pre-trained vision language models by exploiting the semantic and syntactic, better disentangling action and perception, and producing control parameters for given manipulation primitives. [98] implemented a transformer-based Neural Engine that takes the problem and domain in symbolic representation (in PDDL) as input and generates the sequence of actions to solve the problem.

These studies are included in this survey as the Neural Engine takes goals represented as symbols and can generate intermediate steps in symbolic form. On the other hand, these studies do not benefit from Symbol Engines, therefore the generated plans are neither explainable nor verifiable.

[99] learned graph neural network (GNN), whose nodes encode task and domain-related entities, such as objects and outcomes, to discover rules from demonstrations. The long-horizon planning was performed using a gradient-based heuristic, which does not use symbolic knowledge. However, interestingly, in order to add interpretability, they determined the importance of neighboring nodes in decision-making and allowed explanations such as "this node was selected because of its connection with this and this nodes; the most relevant feature being this particular object feature". [100] proposed a developmental progression for symbolic sub-goal discovery in a hierarchical RL framework that combines together the states that have similarities for the given tasks. [101], [102] extended this work to learn both spatial and temporal goal symbols. Focusing on the reachability problem in a mobile robot, a high-level agent finds regions in the reachability-aware goal space, and other agents select the sub-goal symbols for reaching goals and learn how to execute the corresponding actions, increasing the learning speed and scalability.

w/ Non-Neural ML

Here, we review the studies that do not explicitly use generic Neural Engines but benefit from various Machine Learning techniques to learn/process symbols to make inferences and plans with Symbol Engines. They do not use neural networks and historically appeared earlier than the studies that we reviewed so far. Still, we would like to include these studies as they paved the way for the Neuro-Symbolic Robotics, and we review them here as their neural counterparts have already been categorized above.

Initial studies learned sub-symbolic structures that were useful in planning. In a seminal work by [103], the interaction experience of a mobile robot is used to cluster low-level sensory data into categories through self-organizing maps. The system made plans by predicting the next sensory state, where each state was represented by one of the found clusters. Similarly, [104], [105] first applied clustering in the effect space in manipulation and mobile manipulation domains, finding effect categories and learned SVM classifiers that map environment features to effect categories, effectively forming action-effect predictors that were used for planning via tree-search algorithms. [106] learned discrete representations from environment features and a set of predictive models based on these discrete environment symbols. The predictive model is represented with dynamic Bayesian networks (DBNs), which were converted into symbolic plans to generate and execute a sequence of actions. In these studies, symbols were discovered through unsupervised interaction of the robot, similar to the studies in category A.2 (Symbol Discovery), but without Neural Engines.

[24], [107], [108] discussed that high-level planning can be achieved by learning discrete symbols that are used to encode the preconditions and the post-conditions of the action repertoire of agents. Again, as a precursor of neural-network-based approaches, they learned symbols to encode action preconditions and postconditions, and used them in the operators for building the PDDL description of the environment of an agent. [109] extended the previous work with parametrized motor skills, where a robot also learned how to parameterize a skill. [110] proposed a method to form symbols from raw images (pre-processing the image with independent component analysis and then applying Support Vector Machines for image to symbol mapping), following [24] and encoding the given robot skills in Linear Temporal Logic (LTL), enabling symbolic planning for tasks written as LTL formulas. In these works, the state of each object was represented with a fixed-sized vector, assuming that the number of objects was the same across different environments and tasks. [111] extended this approach by using an agent-centric (instead of object-centric) encoding, allowing the discovered symbols to be transferred across different environments. In follow-up work, [112] showed that generalization capability can be increased through use of object-centric representations. [113], [114] also learned object-centric symbols used in preconditions and effects of PDDL operators. Commonly, in order to find discrete representation, the first step in these studies was to apply clustering on the observed interaction instances. As such, the quality of the learned symbols relied on the quality of the state-space partitioning, which was an unsupervised process with no guarantees on the latter planning performance.

[115] proposed an RL-based symbol learning framework where learned symbolic relational abstractions are used for encoding transition and reward model, action effect prediction, and finally, multi-step planning. The optimization of symbol learning focuses not only on enhancing effect prediction performance but also on maximizing rewards. The nearest Neighbor method was used to learn feature-symbol mapping in this work. [116] learned preconditions of manipulation and navigation operators by leveraging the distinction between spatial and non-spatial state variables. Given a collection of adaptive manipulation abstractions, they applied DBSCAN clustering for skill clustering [117], and SVM classifier to map state features to symbols to support planning in mobile manipulation domains. The independence assumption between manipulation and navigation operators allows planning using only manipulation skills and then filling out the navigation steps automatically.

CONCLUSION

In this paper, we reviewed the studies in the recently emerging field of Neuro-Symbolic Robotics. We offered a taxonomy of these studies that categorized them based on how the role of the Neural and Symbol Engines in the respective robotic architectures and how they interplay with each other. While there has been significant effort in non-robotics neuro-symbolic agent architectures [1], [2], [5], [6], [118]–[120], we concluded that its robotics counter-part is in its initial stages.

We also note that while each category addresses different challenges, such as the Neural Engine discovering or learning the symbols used by the Symbol Engine, combining outputs of Neural and Symbol Engines for more robust control, or translating Neural Engines to Symbol Engines to provide verifiability and interpretability, an integrated robotic architecture that fully utilized the benefits of both neural and symbol systems has yet to be introduced.

REFERENCES

[1] A. d'Avila Garcez and L. C. Lamb, "Neurosymbolic ai: The 3rd wave," *arXiv e-prints*, pp. arXiv–2012, 2020.

[2] T. R. Besold, A. d'Avila Garcez, S. Bader, H. Bowman, P. Domingos, P. Hitzler, K.-U. Kühnberger, L. C. Lamb, P. M. V. Lima, L. de Penning *et al.*, "Neural-symbolic learning and reasoning: A survey and interpretation 1," in *Neuro-Symbolic Artificial Intelligence: The State of the Art*. IOS press, 2021, pp. 1–51.

[3] G. Marcus, "The next decade in ai: four steps towards robust artificial intelligence," *arXiv preprint arXiv:2002.06177*, 2020.

[4] J. Achiam, S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F. L. Aleman, D. Almeida, J. Altenschmidt, S. Altman, S. Anadkat *et al.*, "Gpt-4 technical report," *arXiv preprint arXiv:2303.08774*, 2023.

[5] P. Hitzler, A. Eberhart, M. Ebrahimi, M. K. Sarker, and L. Zhou, "Neuro-symbolic approaches in artificial intelligence," *National Science Review*, vol. 9, no. 6, p. nwac035, 2022.

[6] M. K. Saker, L. Zhou, A. Eberhart, and P. Hitzler, "Neuro-symbolic artificial intelligence: Current trends," *Ai Communications*, vol. 34, no. 3, pp. 197–209, 2022.

[7] V. Cohen, J. X. Liu, R. Mooney, S. Tellex, and D. Watkins, "A survey of robotic language grounding: Tradeoffs between symbols and embeddings," 2024. [Online]. Available: https://arxiv.org/abs/2405.13245

[8] S. Harnad, "The symbol grounding problem," *Physica D: Nonlinear Phenomena*, vol. 42, no. 1–3, p. 335–346, Jun. 1990. [Online]. Available: http://dx.doi.org/10.1016/0167-2789(90)90087-6

[9] B. Alt, J. Dvorak, D. Katic, R. Jäkel, M. Beetz, and G. Lanza, "Bansai: Towards bridging the ai adoption gap in industrial robotics with neurosymbolic programming," *arXiv preprint arXiv:2404.13652*, 2024.

[10] A. Gomaa, B. Mahdy, N. Kleer, M. Feld, F. Kirchner, and A. Krüger, "Teach me how to learn: A perspective review towards user-centered neuro-symbolic learning for robotic surgical systems," *arXiv preprint arXiv:2307.03853*, 2023.

[11] N. Hemken, F. Jacob, F. Peller-Konrad, R. Kartmann, T. Asfour, and H. Hartenstein, "How to raise a robot–a case for neuro-symbolic ai in constrained task planning for humanoid assistive robots," *arXiv preprint arXiv:2312.08820*, 2023.

[12] M. Ghallab, A. Howe, C. Knoblock, D. McDermott, A. Ram, M. Veloso, D. Weld, and Wilkins, "Pddl – the planning domain definition language," *Yale Center for Computational Vision and Control Technical Report*, 1998.

[13] K. Mourao, R. Petrick, and M. Steedman, "Using kernel perceptrons to learn action effects for planning," in *Proceedings of the International Conference on Cognitive Systems (CogSys 2008)*, 2008, pp. 45–50.

[14] W. Yuan, C. Paxton, K. Desingh, and D. Fox, "SORNet: Spatial object-centric representations for sequential manipulation," in *Conference on Robot Learning*. PMLR, 2022, pp. 148–157.

[15] T. Migimatsu and J. Bohg, "Grounding predicates through actions," in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 3498–3504.

[16] M. Diehl, C. Paxton, and K. Ramirez-Amaro, "Automated generation of robotic planning domains from observations," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2021, pp. 6732–6738.

[17] M. Helmert, "The fast downward planning system," *Journal of Artificial Intelligence Research*, vol. 26, pp. 191–246, 2006.

[18] S. R. Ahmadzadeh, A. Paikan, F. Mastrogiovanni, L. Natale, P. Kormushev, and D. G. Caldwell, "Learning symbolic representations of actions from human demonstrations," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2015, pp. 3801–3808.

[19] S. Mukherjee, C. Paxton, A. Mousavian, A. Fishman, M. Likhachev, and D. Fox, "Reactive long horizon task execution via visual skill and precondition models," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2021, pp. 5717–5724.

[20] S. Gugliermo, E. Schaffernicht, C. Koniaris, and F. Pecora, "Learning behavior trees from planning experts using decision tree and logic factorization," *IEEE Robotics and Automation Letters*, vol. 8, no. 6, pp. 3534–3541, 2023.

[21] V. Sarathy, D. Kasenberg, S. Goel, J. Sinapov, and M. Scheutz, "Spotter: Extending symbolic planning operators through targeted reinforcement learning," *arXiv preprint arXiv:2012.13037*, 2020.

[22] F. S. Lay, A. S. Bauer, A. Albu-Schäffer, F. Stulp, and D. Leidner, "Unsupervised symbol emergence for supervised autonomy using multimodal latent dirichlet allocations," *Advanced Robotics*, vol. 36, no. 1-2, pp. 71–84, 2022.

[23] A. Ahmetoglu, M. Y. Seker, J. Piater, E. Oztop, and E. Ugur, "DeepSym: Deep symbol generation and rule learning for planning from unsupervised robot interaction," *Journal of Artificial Intelligence Research*, vol. 75, pp. 709–745, 2022.

[24] G. Konidaris, L. P. Kaelbling, and T. Lozano-Perez, "From skills to symbols: Learning symbolic representations for abstract high-level planning," *Journal of Artificial Intelligence Research*, vol. 61, pp. 215–289, 2018.

[25] T. Silver, R. Chitnis, J. Tenenbaum, L. P. Kaelbling, and T. Lozano-Pérez, "Learning symbolic operators for task and motion planning," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2021, pp. 3182–3189.

[26] N. Shah, J. Nagpal, P. Verma, and S. Srivastava, "From reals to logic and back: Inventing symbolic vocabularies, actions and models for planning from raw data," *arXiv preprint arXiv:2402.11871*, 2024.

[27] T. Taniguchi, T. Nagai, T. Nakamura, N. Iwahashi, T. Ogata, and H. Asoh, "Symbol emergence in robotics: a survey," *Advanced Robotics*, vol. 30, no. 11-12, pp. 706–728, 2016.

[28] T. Nakamura, T. Nagai, and N. Iwahashi, "Grounding of word meanings in multimodal concepts using lda," in *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2009, pp. 3943–3948.

[29] J. Nishihara, T. Nakamura, and T. Nagai, "Online algorithm for robots to learn object concepts and language model," *IEEE Transactions on Cognitive and Developmental Systems*, vol. 9, no. 3, pp. 255–268, 2016.

[30] T. Nakamura, T. Nagai, and N. Iwahashi, "Multimodal object categorization by a robot," in *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2007, pp. 2415–2420.

[31] T. Araki, T. Nakamura, T. Nagai, S. Nagasaka, T. Taniguchi, and N. Iwahashi, "Online learning of concepts and words using multimodal LDA and hierarchical Pitman-Yor language model," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2012, pp. 1623–1630.

[32] R. Kuniyasu, T. Nakamura, T. Taniguchi, and T. Nagai, "Robot concept acquisition based on interaction between probabilistic and deep generative models," *Frontiers in Computer Science*, vol. 3, p. 618069, 2021.

[33] K. Furukawa, A. Taniguchi, Y. Hagiwara, and T. Taniguchi, "Symbol emergence as inter-personal categorization with head-to-head latent word," in *2022 IEEE International Conference on Development and Learning (ICDL)*. IEEE, 2022, pp. 60–67.

[34] O. B. Ozturkcu, E. Ugur, and E. Oztop, "High-level representations through unconstrained sensorimotor learning," in *2020 Joint IEEE 10th International Conference on Development and Learning and Epigenetic Robotics (ICDL-EpiRob)*. IEEE, 2020, pp. 1–6.

[35] N. Gopalan, E. Rosen, G. Konidaris, and S. Tellex, "Simultaneously learning transferable symbols and language groundings from perceptual data for instruction following," *Robotics: Science and Systems XVI*, 2020.

[36] A. Ahmetoglu, B. Celik, E. Oztop, and E. Ugur, "Discovering predictive relational object symbols with symbolic attentive layers," *IEEE Robotics and Automation Letters*, 2024.

[37] A. Ahmetoğlu, E. Öztop, and E. Uğur, "Deep multi-object symbol learning with self-attention based predictors," in *2023 31st Signal Processing and Communications Applications Conference (SIU)*. IEEE, 2023, pp. 1–4.

[38] A. Ahmetoglu, E. Oztop, and E. Ugur, "Symbolic manipulation planning with discovered object and relational predicates," *IEEE Robotics and Automation Letters*, 2025.

[39] R. Veerapaneni, J. D. Co-Reyes, M. Chang, M. Janner, C. Finn, J. Wu, J. Tenenbaum, and S. Levine, "Entity abstraction in visual model-based

reinforcement learning," in *Conference on Robot Learning*. PMLR, 2020, pp. 1439–1456.

[40] E. Ugur, Y. Nagai, E. Sahin, and E. Oztop, "Staged development of robot skills: Behavior formation, affordance learning and imitation with motionese," *IEEE Transactions on Autonomous Mental Development*, vol. 7, no. 2, pp. 119–139, 2015.

[41] R. Chitnis, T. Silver, J. B. Tenenbaum, T. Lozano-Perez, and L. P. Kaelbling, "Learning neuro-symbolic relational transition models for bilevel planning," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2022, pp. 4166–4173.

[42] T. Silver, A. Athalye, J. B. Tenenbaum, T. Lozano-Perez, and L. P. Kaelbling, "Learning neuro-symbolic skills for bilevel planning," in *Conference on Robot Learning (CoRL)*, 2022.

[43] J. Achterhold, M. Krimmel, and J. Stueckler, "Learning temporally extended skills in continuous domains as symbolic actions for planning," in *Conference on Robot Learning*. PMLR, 2023, pp. 225–236.

[44] Z. Wang, C. R. Garrett, L. P. Kaelbling, and T. Lozano-Pérez, "Active model learning and diverse action sampling for task and motion planning," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 4107–4114.

[45] ——, "Learning compositional models of robot skills for task and motion planning," *The International Journal of Robotics Research*, vol. 40, no. 6-7, pp. 866–894, 2021.

[46] C. R. Garrett, T. Lozano-Pérez, and L. P. Kaelbling, "Sample-based methods for factored task and motion planning," in *Robotics: Science and Systems*, 2017.

[47] ——, "PDDLStream: Integrating symbolic planners and blackbox samplers via optimistic adaptive planning," in *Proceedings of the international conference on automated planning and scheduling*, vol. 30, 2020, pp. 440–448.

[48] H. Aktas and E. Ugur, "Vq-cnmp: Neuro-symbolic skill learning for bi-level planning," *arXiv preprint arXiv:2410.10045*, 2024.

[49] T. Silver, R. Chitnis, N. Kumar, W. McClinton, T. Lozano-Pérez, L. Kaelbling, and J. B. Tenenbaum, "Predicate invention for bilevel planning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 37, no. 10, 2023, pp. 12 120–12 129.

[50] N. Kumar, W. McClinton, R. Chitnis, T. Silver, T. Lozano-Pérez, and L. P. Kaelbling, "Learning efficient abstract planning models that choose what to predict," in *7th Annual Conference on Robot Learning*, 2023.

[51] A. Li and T. Silver, "Embodied active learning of relational state abstractions for bilevel planning," *arXiv preprint arXiv:2303.04912*, 2023.

[52] L. Wong, J. Mao, P. Sharma, Z. S. Siegel, J. Feng, N. Korneev, J. B. Tenenbaum, and J. Andreas, "Learning adaptive planning representations with natural language guidance," *arXiv preprint arXiv:2312.08566*, 2023.

[53] R. S. Sutton, D. Precup, and S. Singh, "Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning," *Artificial intelligence*, vol. 112, no. 1-2, pp. 181–211, 1999.

[54] G. Konidaris and A. Barto, "Skill discovery in continuous reinforcement learning domains using skill chaining," *Advances in neural information processing systems*, vol. 22, 2009.

[55] P.-L. Bacon, J. Harb, and D. Precup, "The option-critic architecture," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 31, no. 1, 2017.

[56] A. Bagaria and G. Konidaris, "Option discovery using deep skill chaining," in *International Conference on Learning Representations*, 2019.

[57] A. Bagaria, J. K. Senthil, and G. Konidaris, "Skill discovery for exploration and planning using deep skill graphs," in *International Conference on Machine Learning*. PMLR, 2021, pp. 521–531.

[58] N. Shah, "Learning neuro-symbolic abstractions for robot planning and learning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 38, no. 21, 2024, pp. 23 417–23 418.

[59] ——, "Autonomously learning world-model representations for efficient robot planning," Arizona State University, Tech. Rep., 2024.

[60] D. Molina, K. Kumar, and S. Srivastava, "Learn and link: Learning critical regions for efficient planning," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 10 605–10 611.

[61] N. Shah and S. Srivastava, "Using deep learning to bootstrap abstractions for hierarchical robot planning," *arXiv preprint arXiv:2202.00907*, 2022.

[62] Y. Liang, N. Kumar, H. Tang, A. Weller, J. B. Tenenbaum, T. Silver, J. F. Henriques, and K. Ellis, "Visualpredicator: Learning abstract world models with neuro-symbolic predicates for robot planning," *arXiv preprint arXiv:2410.23156*, 2024.

[63] A. Athalye, N. Kumar, T. Silver, Y. Liang, T. Lozano-Pérez, and L. P. Kaelbling, "Predicate invention from pixels via pretrained vision-language models," *arXiv preprint arXiv:2501.00296*, 2024.

[64] E. Saccon, A. Tikna, D. De Martini, E. Lamon, L. Palopoli, and M. Roveri, "When prolog meets generative models: a new approach for managing knowledge and planning in robotic applications," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 17 065–17 071.

[65] A. Capitanelli and F. Mastrogiovanni, "A framework for neurosymbolic robot action planning using large language models," *Frontiers in Neurorobotics*, vol. 18, p. 1342786, 2024.

[66] M. Han, Y. Zhu, S.-C. Zhu, Y. N. Wu, and Y. Zhu, "Interpret: Interactive predicate learning from language feedback for generalizable task planning," *arXiv preprint arXiv:2405.19758*, 2024.

[67] J. Sun, H. Sun, T. Han, and B. Zhou, "Neuro-symbolic program search for autonomous driving decision module design," in *Conference on Robot Learning*. PMLR, 2021, pp. 21–30.

[68] M. Ahn, A. Brohan, N. Brown, Y. Chebotar, O. Cortes, B. David, C. Finn, C. Fu, K. Gopalakrishnan, K. Hausman *et al.*, "Do as i can, not as i say: Grounding language in robotic affordances," *arXiv preprint arXiv:2204.01691*, 2022.

[69] A. Chowdhery, S. Narang, J. Devlin, M. Bosma, G. Mishra, A. Roberts, P. Barham, H. W. Chung, C. Sutton, S. Gehrmann *et al.*, "Palm: Scaling language modeling with pathways," *Journal of Machine Learning Research*, vol. 24, no. 240, pp. 1–113, 2023.

[70] G. Tziafas and H. Kasaei, "Enhancing interpretability and interactivity in robot manipulation: A neurosymbolic approach," *arXiv preprint arXiv:2210.00858*, 2022.

[71] I. Sharifi, M. Yildirim, and S. Fallah, "Towards safe autonomous driving policies using a neuro-symbolic deep reinforcement learning approach," *arXiv preprint arXiv:2307.01316*, 2023.

[72] C. Cornelio and M. Diab, "Recover: A neuro-symbolic framework for failure detection and recovery," *arXiv preprint arXiv:2404.00756*, 2024.

[73] G. Chalvatzaki, A. Younes, D. Nandha, A. T. Le, L. F. Ribeiro, and I. Gurevych, "Learning to reason over scene graphs: a case study of finetuning gpt-2 into a robot language model for grounded task planning," *Frontiers in Robotics and AI*, vol. 10, p. 1221739, 2023.

[74] Y. Zhu, J. Tremblay, S. Birchfield, and Y. Zhu, "Hierarchical planning for long-horizon manipulation with geometric and symbolic scene graphs," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 6541–6548.

[75] A. M. Wells, N. T. Dantam, A. Shrivastava, and L. E. Kavraki, "Learning feasibility for task and motion planning in tabletop environments," *IEEE robotics and automation letters*, vol. 4, no. 2, pp. 1255–1262, 2019.

[76] D. Driess, J.-S. Ha, and M. Toussaint, "Deep visual reasoning: Learning to predict action sequences for task and motion planning from an initial scene image," *arXiv preprint arXiv:2006.05398*, 2020.

[77] R. Chitnis, D. Hadfield-Menell, A. Gupta, S. Srivastava, E. Groshev, C. Lin, and P. Abbeel, "Guided search for task and motion plans using learned heuristics," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2016, pp. 447–454.

[78] T. Hiraoka, T. Onishi, T. Imagawa, and Y. Tsuruoka, "Refining manually-designed symbol grounding and high-level planning by policy gradients," *arXiv preprint arXiv:1810.00177*, 2018.

[79] L. Lamanna, A. Saetti, L. Serafini, A. Gerevini, P. Traverso *et al.*, "Online learning of action models for pddl planning." in *IJCAI*, 2021, pp. 4112–4118.

[80] L. Lamanna, M. Faridghasemnia, A. Gerevini, A. Saetti, A. Saffiotti, L. Serafini, P. Traverso *et al.*, "Learning to act for perceiving in partially unknown environments." in *IJCAI*, 2023, pp. 5485–5493.

[81] A. Ahmetoglu, E. Ugur, M. Asada, and E. Oztop, "High-level features for resource economy and fast learning in skill transfer," *Advanced Robotics*, vol. 36, no. 5-6, pp. 291–303, 2022.

[82] S. Cheng and D. Xu, "League: Guided skill learning and abstraction for long-horizon manipulation," *IEEE Robotics and Automation Letters*, 2023.

[83] U. Rakhman, J. Ahn, and C. Nam, "Fully automatic data collection for neuro-symbolic task planning for mobile robot navigation," in *2021 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE, 2021, pp. 450–455.

[84] L. Lamanna, L. Serafini, M. Faridghasemnia, A. Saffiotti, A. Saetti, A. Gerevini, and P. Traverso, "Planning for learning object properties," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 37, no. 10, 2023, pp. 12 005–12 013.

[85] P. Verma, S. R. Marpally, and S. Srivastava, "Discovering user-interpretable capabilities of black-box planning agents," in *Proceedings of the 19th International Conference on Principles of Knowledge Representation and Reasoning*, 2022.

[86] O. Bastani, Y. Pu, and A. Solar-Lezama, "Verifiable reinforcement learning via policy extraction," *Advances in neural information processing systems*, vol. 31, 2018.

[87] G. Weiss, Y. Goldberg, and E. Yahav, "Extracting automata from recurrent neural networks using queries and counterexamples," in *International Conference on Machine Learning*. PMLR, 2018, pp. 5247–5256.

[88] I. C. Kaadoud, N. P. Rougier, and F. Alexandre, "Knowledge extraction from the learning of sequences in a long short term memory (lstm) architecture," *Knowledge-Based Systems*, vol. 235, p. 107657, 2022.

[89] S. Carr, N. Jansen, and U. Topcu, "Verifiable rnn-based policies for pomdps under temporal logic constraints," *arXiv preprint arXiv:2002.05615*, 2020.

[90] X. Sun and Y. Shoukry, "Neurosymbolic motion and task planning for linear temporal logic tasks," *IEEE Transactions on Robotics*, 2024.

[91] G. Anderson, A. Verma, I. Dillig, and S. Chaudhuri, "Neurosymbolic reinforcement learning with formally verified exploration," *Advances in neural information processing systems*, vol. 33, pp. 6172–6183, 2020.

[92] K. Namasivayam, H. Singh, V. Bindal, A. Tuli, V. Agrawal, R. Jain, P. Singla, and R. Paul, "Learning neuro-symbolic programs for language guided robot manipulation," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 7973–7980.

[93] J. Mao, C. Gan, P. Kohli, J. B. Tenenbaum, and J. Wu, "The neuro-symbolic concept learner: Interpreting scenes, words, and sentences from natural supervision," in *International Conference on Learning Representations*, 2019. [Online]. Available: https://openreview.net/forum?id=rJgMlhRctm

[94] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, J. Dabis, C. Finn, K. Gopalakrishnan, K. Hausman, A. Herzog, J. Hsu *et al.*, "Rt-1: Robotics transformer for real-world control at scale," *arXiv preprint arXiv:2212.06817*, 2022.

[95] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, X. Chen, K. Choromanski, T. Ding, D. Driess, A. Dubey, C. Finn *et al.*, "Rt-2: Vision-language-action models transfer web knowledge to robotic control," *arXiv preprint arXiv:2307.15818*, 2023.

[96] A. O'Neill, A. Rehman, A. Gupta, A. Maddukuri, A. Gupta, A. Padalkar, A. Lee, A. Pooley, A. Gupta, A. Mandlekar *et al.*, "Open x-embodiment: Robotic learning datasets and rt-x models," *arXiv preprint arXiv:2310.08864*, 2023.

[97] R. Wang, J. Mao, J. Hsu, H. Zhao, J. Wu, and Y. Gao, "Programmatically grounded, compositionally generalizable robotic manipulation," *arXiv preprint arXiv:2304.13826*, 2023.

[98] V. Pallagani, B. Muppasani, B. Srivastava, F. Rossi, L. Horesh, K. Murugesan, A. Loreggia, F. Fabiano, R. Joseph, Y. Kethepalli *et al.*, "Plansformer tool: Demonstrating generation of symbolic plans using transformers." in *IJCAI*, 2023, pp. 7158–7162.

[99] Y. Lin, A. S. Wang, E. Undersander, and A. Rai, "Efficient and interpretable robot manipulation with graph neural networks," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 2740–2747, 2022.

[100] M. Zadem, S. Mover, and S. M. Nguyen, "Goal space abstraction in hierarchical reinforcement learning via reachability analysis," *arXiv preprint arXiv:2309.07168*, 2023.

[101] ——, "Reconciling spatial and temporal abstractions for goal representation," *ICLR*, 2024.

[102] M. Zadem, S. Mover *et al.*, "Emergence of a symbolic goal representation with an intelligent tutoring system based on intrinsic motivation," in *Intrinsically-Motivated and Open-Ended Learning Workshop@ NeurIPS2023*, 2023, pp. 423–428.

[103] J. Pisokas and U. Nehmzow, "Experiments in subsymbolic action planning with mobile robots," in *Symposium on Adaptive Agents and Multi-agent Systems*. Springer, 2003, pp. 216–229.

[104] E. Ugur, E. Sahin, and E. Öztop, "Affordance learning from range data for multi-step planning." in *EpiRob*, 2009.

[105] E. Ugur, E. Şahin, and E. Oztop, "Unsupervised learning of object affordances for planning in a mobile manipulation platform," in *2011 IEEE International Conference on Robotics and Automation*. IEEE, 2011, pp. 4312–4317.

[106] J. Mugan and B. Kuipers, "Autonomous learning of high-level states and actions in continuous environments," *IEEE Transactions on Autonomous Mental Development*, vol. 4, no. 1, pp. 70–86, 2011.

[107] G. Konidaris, L. Kaelbling, and T. Lozano-Perez, "Constructing symbolic representations for high-level planning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 28, no. 1, 2014.

[108] G. Konidaris, L. P. Kaelbling, and T. Lozano-Perez, "Symbol acquisition for probabilistic high-level planning," in *AAAI Press/International Joint Conferences on Artificial Intelligence*, 2015.

[109] B. Ames, A. Thackston, and G. Konidaris, "Learning symbolic representations for planning with parameterized skills," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 526–533.

[110] A. Pacheck, S. James, G. Konidaris, and H. Kress-Gazit, "Automatic encoding and repair of reactive high-level tasks with learned abstract representations," *The International Journal of Robotics Research*, vol. 42, no. 4-5, pp. 263–288, 2023.

[111] S. James, B. Rosman, and G. Konidaris, "Learning portable representations for high-level planning," in *International Conference on Machine Learning*. PMLR, 2020, pp. 4682–4691.

[112] ——, "Autonomous learning of object-centric abstractions for high-level planning," in *International Conference on Learning Representations*, 2022.

[113] E. Ugur and J. Piater, "Bottom-up learning of object categories, action effects and logical rules: From continuous manipulative exploration to symbolic planning," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2015, pp. 2627–2633.

[114] ——, "Refining discovered symbols with multi-step interaction experience," in *2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*. IEEE, 2015, pp. 1007–1012.

[115] N. Jetchev, T. Lang, and M. Toussaint, "Learning grounded relational symbols from continuous data for abstract reasoning," in *Proceedings of the 2013 ICRA Workshop on Autonomous Learning*, 2013.

[116] E. Rosen, S. James, S. Orozco, V. Gupta, M. Merlin, S. Tellex, and G. Konidaris, "Synthesizing navigation abstractions for planning with portable manipulation skills," in *Conference on Robot Learning*, 2023, pp. 2278–2287.

[117] M. Ester, H.-P. Kriegel, J. Sander, X. Xu *et al.*, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *kdd*, vol. 96, no. 34, 1996, pp. 226–231.

[118] G. Marcus, "The next decade in ai: four steps towards robust artificial intelligence," *arXiv preprint arXiv:2002.06177*, 2020.

[119] A. d. Garcez, T. R. Besold, L. De Raedt, P. Földiak, P. Hitzler, T. Icard, K.-U. Kühnberger, L. C. Lamb, R. Miikkulainen, and D. L. Silver, "Neural-symbolic learning and reasoning: contributions and challenges," in *2015 AAAI Spring Symposium Series*, 2015.

[120] P. Hitzler and M. K. Sarker, *Neuro-symbolic artificial intelligence: The state of the art*. IOS press, 2022.